Django projects have a specific folder and file structure that is designed to organize code, templates, static files, and configuration settings. Understanding this structure is crucial for effective Django development. Here's an overview of the main components of a typical Django project:

Django Project Folder Structure:

```
project_name/         # Main project directory
│
├── project_name/      # Actual Django project module
│   ├── __init__.py
│   ├── asgi.py        # ASGI configuration (for async support)
│   ├── settings.py    # Django settings for the project
│   ├── urls.py        # Main URL routing configuration
│   └── wsgi.py        # WSGI configuration (for deployment)
│
├── app_name/          # Django app(s) directory (can have multiple apps)
│   ├── migrations/    # Database migration files
│   ├── __init__.py
│   ├── admin.py       # Admin site configuration
│   ├── apps.py        # App-specific configuration
│   ├── models.py      # Database models
│   ├── tests.py       # Unit tests
│   └── views.py       # Views for the app
│
├── static/            # Static files (CSS, JavaScript, images, etc.)
│
├── templates/         # HTML templates
│
├── media/             # User-uploaded media files (if applicable)
│
├── manage.py          # Command-line utility for managing Django projects
│
└── venv/              # Virtual environment (optional but recommended)
```

Project-Level Files:

- **`manage.py`:** A command-line utility for interacting with the Django project. It's used for running development servers, creating database tables, and other management tasks.

- **`project_name/`:** This is the Python package containing the main settings for your Django project. It also includes configuration files for ASGI (`asgi.py`), WSGI (`wsgi.py`), and URL routing (`urls.py`).

App-Level Files:

- **`models.py`:** Defines the data models for the app, specifying the structure of the database tables.

- **`views.py`:** Contains the views (functions or classes) that handle HTTP requests and return responses.

- **`urls.py`:** Specifies the URL patterns for the app, mapping URLs to corresponding views.

- **`admin.py`:** Configures the Django admin interface to manage models from the app.

- **`apps.py`:** Configuration file for the app, where you can define app-specific settings.

- **`migrations/`:** Directory for storing database migration files. These files manage changes to the database schema over time.

Other Directories:

- **`static/`:** Holds static files like CSS, JavaScript, and images that are served directly by the web server.

- **`templates/`:** Contains HTML templates used by the views to generate dynamic content.

- **`media/`:** Typically used for storing user-uploaded media files (images, documents, etc.), if your application handles file uploads.

- **`venv/`:** The virtual environment directory (if you choose to use one for isolation of dependencies).

Understanding and adhering to this structure makes it easier to collaborate with other developers, manage codebase growth, and follow Django best practices. Each app within the project follows a similar structure, promoting modularity and maintainability.